ISSN 0112-3351

BEEBLET

CONTENTS

Editorial 2
Beginners' Column 3
President's Report 6
Classifieds 8
My Program will ... er.. 9
Illustrations Required 10
Message to Younger Readers 10
Service Agents and Dealers 11
Build your own Database Manager -Pt2 12
CONTEST 19
Electron Users E2P Second Processor 20
On Creatively Using Shadows.. 20
A TV of Many Colours 23
For Sale 24
News from Nelson 25
We Know How he Feels! 26
HELP 27
Hall of Fame - with a difference 29
Games Menu Label Printer 30
Master Matters 34
Auckland Bulletin Board 39
Membership 39
User Group Meetings 40
Beeblet- Contributions and Deadlines 40

Volume 4 Number 9 OCTOBER 1986

BBC / ACORN

COMPUTER USER GROUP NZ (inc)

EDITORIAL

As you were warned last month (P38), Bruce Wills is giving up the position as Editor of *Beeblet*.

On behalf of you all, I wish to record our grateful thanks to Bruce for all the effort he has put in over the last few years. You may not be aware, but Bruce has given whole weekends every month to preparing the magazine for publishing, and on many occasions, when articles, reviews and programs have not come in by the deadline, he has made significant original input of his own.

Beeblet is the most significant financial outlay of the User Group; we all have a great debt to Bruce for all the work he has done.

As a measure of the effort required to produce the magazine, a Gang of Three has been volunteered to continue the publication. Debbie Cosgrove and Steve Rose will be performing an initial editorial role, eg: checking that programs submitted do in fact do as their authors claim, and Graham Ellett finalising the layout preparatory to printing.

We will be looking to expanding the Gang to spread the checking load, and to provide some direct contacts to encourage items for the Beeblet. In particular, we would like to encourage some of the younger Beeb users to make contributions.

The best method of submitting items is on disc or tape, with notes, reviews or articles as Wordwise or View files. Control characters should be minimised, with a separate printout marked up to indicate any preferred layout if desired. Discs or tapes will be returned, and if the authors were to indicate any programs that have been published in the *Beeblet* that they would like, every effort will be made to satisfy the request.

Reviews or articles (but not programs) in written or printed form are acceptable, although Wordwise files would be preferred.

In any case, make sure you add a note clearly indicating your name, address and telephone number, and any instructions required. Send all contributions to:

BEEBLET
The BBC/Acorn User Group of NZ INC.,
P O BOX 9592
WELLINGTON

by the deadline indicated on the back page of the Beeblet.

Remember, *Beeblet* is your magazine. It is only as good as you make it.

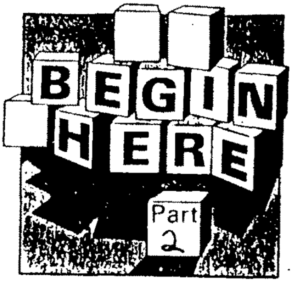
Having got that off our chests - on with this month's magazine.......

Debbie Cosgrove
Steve Rose
Graham Ellett

October 1986

BEGINNERS' COLUMN

By Neville Hudson



INTRODUCTION

I hope last months article was of some benefit to those of you who still call yourselves beginners to computing. Remember if there is something you would like more information on or something which hasn't been covered and you would like it to be covered then drop me a line. Please address all correspondence relating to these articles to Beginners' Column. P.O. Box 9592, Wellington.

HOW TO MAKE A BACK-UP COPY

I doubt if there is anything more frustrating than accidentally losing a favourite program whether typed in by hand or purchased. Most software houses allow you to make one back-up copy of a disc as a safe guard against accidental loss. You are advised to check on copyright conditions before you copy any software.

Before you start ensure the disc you intend to receive the information on has been formatted and that the original has a write protect TAB on it. If you have a single drive type *ENABLE and press RETURN. This command is a protective measure to prevent accidental copying. Now type *BACKUP00 and press RETURN. This command is telling the computer to make a back-up copy on drive 0 using the disc in drive 0 as the source. Follow the prompts as to when to change over the discs. Take care not to get the discs mixed up while changing them over.

If you have twin disc drives then place the original disc in drive 0 and the new disc in drive 1. Type *ENABLE and press RETURN then type *BACKUP01 and press RETURN.

Once this is complete you will have an exact copy of the original disc. Remember to place a write protect TAB on it as soon as possible.

If the above steps have failed then I am afraid that at this stage you won't be able to make a back-up copy. Someone else in the User Group may be able to help but it is beyond the scope of this series to go any deeper into the making of a back-up copy.

MORE ON USING DISCS

Last month I asked you to type in some homework. Place this disc in drive 0. The menu program should have been saved as !MENU. Type *CAT and check that this is in fact the case. If you have saved the homework as something else then type *RENAME (filename you gave it without brackets) !MENU and press return. eg. *RENAME HOMEWK !MENU This will rename the file as !MENU. If you now type CHAIN"!MENU" and press RETURN a title will appear at the top of the screen and a menu will appear below this. At this stage there is nothing in the menu except the words Title and Details and if you press A you will get an error message.

Press the BREAK key. I want to now move on to how you can use a menu to store your favourite programs ready for quick access. Firstly you will need a program to autostart (Boot) the disc.

Type *BUILD !BOOT and press RETURN. The number 1 will appear below what has been typed.

Type *TV0,1 and press RETURN. A number 2 will now appear. *TV0,1 is a command that helps prevent excessive flicker on the screen. It is particularly useful when using a TV set.

Type CHAIN "!MENU" and press RETURN. A number 3 will appear.

Press the ESCAPE key and the disc drive will start up.

Type *CAT and press RETURN. You should now see that a file called !BOOT has been added to the disc catalogue.

Type *OPT4,3 and press RETURN. This command turns the disc on so it can autostart by using the file called !BOOT. Now the good bit.

Press SHIFT and BREAK at the same time. Take your finger off BREAK first and then take your finger off SHIFT. The disc drive will start up and the menu you typed in for homework will appear. Pressing SHIFT and BREAK together is called booting a disc.

Press BREAK and then type *ACCESS !BOOT L and press RETURN. Now to give the disc a title. Type *TITLE and press SHIFT and function key 1. Continue typing HOMEWORK and then press RETURN. The disc drive will start up.

Now press *CAT and press RETURN. Notice the changes to the disc catalogue title. Pressing different function keys produce other colours. Try this and see. eg. *TITLE f3 HOMEWORK and press RETURN.

The menu can now be used to display programs and to access them more easily. IT IS NOT USER GROUP POLICY TO ENCOURAGE THE COPYING OF PROGRAMS AND IN FACT THE USER GROUP WARNS AGAINST BREAKING COPYRIGHT. However, if you have programs that can be copied, for example Public Domain software, then follow these steps.

HOW TO COPY A SINGLE CATALOGUE FILENAME

Select a program from a disc catalogue and copy down its filename. If you have twin disc drives, place the disc with your selected program into drive 0 and the disc you have been working with up to now into drive 1. Type *COPY01 (filename of selected program) and press RETURN. eg *COPY01 INVADER. If you have only one disc drive then type *COPY00 (filename of selected program) and follow the prompts. WARNING!!! the source disc is the disc with your selected program you want to copy and the destination disc is the one you put the menu on to. Make sure that the source disc has a write protect TAB before you attempt to copy it.

Type LOAD"!MENU" and press RETURN. Type the following:

```
1900 INVADER,Invaders,Arcade game
```

Press RETURN. In this example the selected program has a filename of INVADER. You will need to type in your programs filename followed by its title and then a very brief description about the program. Notice the use of commas after each of the first two parts.

Once you have added your program details type SAVE"!MENU" and press RETURN. Boot in the disc as already described. Press A and your program will be selected.

POINTS ARISING

Before you go mad and put all your programs on menu you should be warned of the following:

1. Many programs have more than one filename which is loaded as it is needed. If you miss one of these then the program will not work.
2. If you load the !MENU program and list it again you will see there is a large gap of numbers between line 1900 and line 3500. You have just put in line 1900 so further additions will need to be added after line 1900. I suggest in steps of 50. DO NOT put anything in at or after line 3500.
3. At this stage do not put more than 15 programs into your menu.
4. Always SAVE the !MENU before running it after making additions.
5. If the menu list doesn't look very good because of the width of the statements leave it for now and I will tell you how to fix it later.
6. Remember the 31 filenames restriction on each disc catalogue when using the Acorn DFS.

HOMEWORK

Continue to experiment with your own menu list.

……

PRESIDENT'S REPORT – OCTOBER 1986

EDITORIAL CHANGES

As you will know from the last *Beeblet*, Bruce Wills has resigned from the editorship and the National Committee. On behalf of the User Group I would like to thank him for the tremendous efforts he has put into the magazine over the last couple of years. A measure of the amount of work involved is that he is being replaced by an editorial committee of three – Debbie Cosgrove, Graham Ellett, and Steve Rose. Both Debbie and Graham have been co-opted onto the National Committee to fill the vacancy created by Bruce's resignation. We welcome them and wish them well in their work.

USER GROUP FUTURE

It is, once again, time for a little navel gazing to determine the role and future of our Group. The survey conducted at the end of last year clearly showed that our most important activity is the production of *Beeblet*. While there is not great cause for concern, the current membership of around 350 is down on last year, which means that, per member, the cost of producing *Beeblet* is higher than it should be.

We have noted the proliferation of local BBC groups around the country, some producing their own local newsletter. Ideally, this material should instead be contributed to *Beeblet* as it is often of interest to the whole country and not just the local region. (I believe that one of *Beeblet's* main roles should be to act as a kind of bulletin board, with all sorts of interesting titbits about what people are doing.) This would reduce costs of running a local group to just the amount needed to rent some venue for holding meetings.

Unfortunately, since inevitably only some members of such a group would already belong to our NZ-wide User Group, it would mean forcing the remainder to join as well if they want to read their local newsletter information. Because it would look like a Wellington-based organization was throwing their weight around we have tended not to push the approach of subsuming such local groups and their newsletters. We have ignored cases where our copyright material from *Beeblet* has been reproduced locally. Furthermore, we publish details of all local group meetings that we know about, whether they are 'one of us' or not, in the belief that we should encourage BBC and Acorn computer owners to get together whether it is under our banner or not.

There is no easy solution to this problem. Perhaps there is a need for some decentralization of our activities – for too long much of the effort has come from the Wellington area. It would be nice to get some other regions to take over some of the jobs involved in running the Group. For instance, with modern communications there is no reason why the magazine couldn't be produced elsewhere. There is also a possibility of using STARNET or MYRIAD for communication amongst Group members around the country; this could be organized by one of the small centres which stands more to gain from access to a national bulletin board service than do the larger centres, such as Auckland, which have local bulletin boards available for a free phone call.

However, less I seem unduly pessimistic, rest assured that there is no likelihood of either the User Group or *Beeblet* going away. I would welcome any opinions on what the role of the National Group should be, and once again urge all of you to encourage other BBC and Electron owners to join us so that we can continue to be a strong, healthy group with a useful magazine at a reasonable price.

BULLETIN BOARDS

On the subject of bulletin boards, there is a new one operated by Murray's Computer Village in Lower Hutt. It is only available outside shop hours - that is, Monday to Saturday morning (except Thursday night) 5:30 pm to 9:00 am; 8:30 pm Thursday to 9:00 am Friday, and 12:30 pm Saturday to 9:00 am Monday. The phone number is (04) 697-432, and use 300 baud to log on. It is so busy that I have yet to log on myself. Good luck.

The Wellington Micro Society is also starting up a bulletin board again - should be running by the end of October. It will be available only to members (the annual subscription is $20), and will offer a 24-hr service. For details, you can contact Peter Cowley on (04) 837-503 or write to P.O. Box 1581, Wellington.

MASTER COMPACT AND MASTER PRICINGS

You read all about the new Master Compact in the last issue of "Beeblet", thanks to Bruce Smith of "Acorn User" who very kindly sent (on disk) an advance copy of his article. I am happy to plug his magazine (and books) in gratitude.

The Compact has now been launched in New Zealand. The cost, with GST, is $2195 without a monitor, $2283 with a VHF modulator so that a television can be used as a monitor, $2508 with a monochrome monitor, and $2888 with a (medium resolution) RGB monitor. Incidentally, the Master-128 has now risen in price to $1995 "to cover increased costs and GST". (It seems hardly fair to blame that rise on GST, given that a 10% wholesale sales tax on computers was replaced by 10% of GST.)

Let's compare the NZ prices with the latest advertised in Britain (from October "Micro User"), without VAT:

	NZ$	Pounds	Ratio
Master-128	1995	395	5.05
Compact without monitor	2195	399	5.50
Compact with VHF modulator	2283	-	-
Compact with monochrome monitor	2508	469	5.34
Compact with RGB monitor	2888	599	4.82

We seem to back in the same old situation of multiplying British prices, in pounds and without VAT, by a factor of five to get prices in NZ dollars, just like it was back in the bad old days of 40% sales taxes.

Now I know that there are a lot of overheads in the business, and that it costs a lot to airfreight computers in. But a ratio of around five to one seems excessive. At worst, in its recent low, it took just three NZ dollars to buy one pound. Once again, it's looking cheaper to mail-order computers from Britain than buy locally. For example, a Master-128 would cost around 470 pounds including air freight, which converts to around $1350 at the current exchange rate of 35p to the NZ$. Add 10% GST when it arrives, and the price is $1485, or $500 less than the retail NZ price.

You will, of course, have to take the risk of no warranty. To give Barsons credit, they have just increased the NZ warranty period from 3 to 12 months.

I guess you just have to pay Rolls Royce prices in NZ for a Rolls Royce machine. The question is, will you?

Neil Gordon

....
.... ::::::::: ::::::::: ::::::::: ::::::::: ....
::::::::: :::::::::

CLASSIFIEDS:

Computers
(They all should fall into that class)

Paul Collins is looking for 32K Shadow Ram Board (Aires of Watford). Replies to 185 Victoria St., Cambridge or ZL1AGB.

Paul Collins (yes the same one) is wanting contact with a PL Graphics Digitiser user. Replies: as above.

I do not have details of the Dunedin group that advertised last month - so I'll use this month's issue to respond:

Andrew Ammundsen advises he has the following for sale:
  Watford ROM/RAM board 32K (with 5 ROM sockets, brand NEW) price $140, and
  Solidisk 64K RAM board (with 2 32K ROM sockets, and with spring loaded clips attached to wires - so no soldering, and with 5 discs of software) Price $130
Address: 48 Kahikatea Drive, Hamilton - phone (071) 434008

J.R.Gleve advises he has a Solidisk 128K RAM for sale complete with 12 discs and Manual. Price $250
Address: 44 Mornington Rd Dunedin. Phone Dunedin 36-668 evenings

Ed: Please note, it is preferred to publish details of buyers/sellers addresses and or phone numbers so that direct contact can be made.

My Program Will...er...sort of...um...

or

How to Get What You Want From Your Program

By Steven Rose

Last month in the HELP column, you may have noticed that one of the Programming Laws said:

> "If you don't know what your program is supposed to do, don't start writing it"

Some people seem to have Moon Watcher's problem (in 2001 a Space Odyssey) when they write a program: I'm not quite sure what to do next, but I'll think of something.

The problem is often that they know what they want, but they're not quite sure how to achieve it.

First, Define the Problem

I've found that the best solution is to switch to the wordprocessor and write a description of the program, as if for a manual: what it does, what it needs (files, a certain disc, Econet, etc), special features, and any restrictions that I can think of. I try to make it as clear as possible, so that a total stranger could understand it, which often causes it to be over a page long. This forces me to clarify my ideas, and gives me something to refer to that's easier than a program to read quickly. It also makes a good starting point for a Beeblet article!

Break it Down into Steps

Now that I know what I want, I start to write what the program will do, as a list of three or four word statements. This is my 'function list'. It contains things like 'Set up variables', 'print keys to use', 'get input file name', 'process file'. Some of these need to be broken down into sublists of functions, which I indent to show the 'parent' function. These functions are the sort of things that you would put into separate procedures. The indented sublists would be procedures called from within the parent procedure.

Turn them into 'Computer World' Steps

While I'm writing this list, I find that my subconscious mind starts to come up with ways to achieve what I want, so I go back and fill in a short description of how each function is going to be performed, next to the appropriate function in the list.

I suppose I'm really using the wordprocessor as an ideas processor, getting my thoughts down as they pop up, then putting them where they belong. There's no reason why you can't use a piece of paper instead, just use two columns (that's what I do at work if I get stumped and the company doesn't have a very good text editor). The advantage of a wordprocessor is that you can insert things into the list easily, so you don't lose your train of thought.

Because the methods of doing the tasks are close together, but in order, I can read down through them to see what happens where. Usually something is incompatible with something else, and sometimes this means that I have to rewrite part of my original description, and/or change the function list.

You may have recognised that I've been describing a structure diagram, but using a list instead of a diagram; I find that a list is more flexible than a diagram.

Now Write the Program!

Now you have a list of what the program will do, and how it will do each thing, so you can write a procedure for each function in the list, add a few REPEAT...UNTIL loops here and there, where a procedure needs to be called repeatedly, and you'll have a program! It may even do what you want!

....

ILLUSTRATIONS

I am sure that you will agree that a sprinkling of illustrations improves any magazine, and *Beeblet* is no exception. I know I will often be looking for items to complete the odd page - like this one, and a folder full of illustrations of general interest to computer buffs would be very handy.

Full page cartoons, like you will find on page 26, are useful but its the small, half page illustration that would be most welcome.

Screendumps can also be used - an example is used to head the article "Master Matters" by Neil Gordon.

....

Special Message to Younger Readers:

That's an ambiguous title. But I know that many of the younger members of User Groups can beat Mum or Dad with their hands tied behind their backs (come to think of it, there could be problems just using the nose) as far as ideas and programs are concerned.

Please share your ideas through *Beeblet* - we, of all ages, have a lot to learn. You might even find that Mum or Dad is only to pleased to assist you finalise your treatise.

What about your ideas as to what *Beeblet* can provide for you?

SERVICE AGENTS AND DEALERS

The NZ network of service and dealers for the BBC is as follows:

AUCKLAND:	Telephone
Sales and Service-	
Barson Computers NZ Ltd,	
1 Ngaire Avenue,	504-049
Auckland 3	504-631
John Gilbert Electronics	
75 Parnell Rise,	
Parnell.	390-839
Other Dealers-	
Computer Terminal	
Hinemoa St	419-053
Neil Gladden,	
Data Base Services,	
8 Kowhatu Rd	
Whitcoulls Ltd-	
186 Queen St	778-329
312 Broadway,Newmarket	543-129
BLENHEIM:	
Whitcoulls Ltd	
30 Market St Nth	88-039
CHRISTCHURCH:	
Sales and Service-	
Aftron Computers Ltd	
161 Barbados St.	797-445
Dealers-	
TVNZ Enterprises Shop,	
P.O. Box 1945	
Whitcoulls Ltd	
111 Cashel St	794-580
Canterbury Centre, High St	50-159
201 Manchester St,	787-561
DUNEDIN:	
Whitcoulls Ltd	
143 George St.	774-120
HAMILTON:	
Whitcoulls Ltd	
175 Victoria St.	82-174

INVERCARGILL:	Telephone
Whitcoulls Ltd	
66 Dee St.	86-063
SOUTHLAND:	
Lyndon McEntee,	
Educomp, PO Box 255,	
Gore.	7365
NEW PLYMOUTH:	
Winston Electronics	
26 Berwyn Ave.	298 7769
TAURANGA:	
Sales and Service-	
Graham Cheer	
Whitcoulls Ltd	
97 Grey St	87-738
TE PUKE:	
May & Associates	
PO Box 345.	83-257
TIMARU:	
Whitcoulls Ltd	
182 Stafford St.	88-185
WANGANUI:	
Whitcoulls Ltd	
148 Victoria Ave.	54-193
WELLINGTON:	
Service-	
APTEC	
(Applied Engineering and Technology)	
3 Halleys Lane	850-811
	850-800
Dealers-	
Whitcoulls Ltd	
Lambton Quay	721-921
Neville Hudson,	
249 Darlington Road,	
Miramar	882-444

BUILD YOUR OWN DATABASE MANAGER.

Beebug Filer - a database for all.

Part 2

We continue our major new software project, BEEBUG'S database program, by describing the file structure in detail and adding functions to provide flexible formatting and output of your records.

Last month Neville Hudson introduced a major software project in the magazine to construct your own database program. This program is a reprint of BEEBUG'S program which was written by Mike Williams. Beebugsoft have kindly agreed to allow us to publish their very successful series on the Beebug Filer. Copyright remains with Beebugsoft.

The basic program, called BEEBUG Filer, was presented in the first instalment together with a general introduction and description. This month we shall look at the fundamentals of our database program in some detail, but first we will see how we can improve on some of our first steps, and add some exciting new features. The major addition will allow you to output your records in a flexible format of your own design.

ERRATUM

Before proceeding further, Neville has advised that there was an error in line 2820 of the listing on Page 15 of last month's *Beeblet*. The line should have read as follows:

```
2820 recs=2*f:FOR I=1 TO f:recs=recs+width%(I):NEXT I
```

Apologies for any difficulties arising. Back to this month's article:

GENERAL DESCRIPTION

The program uses mode 3 to allow the maximum amount of information to be displayed on the screen. We shall have to watch memory space carefully as the program develops, but in the meantime we must endeavour to keep the program as concise as possible and yet provide a clear and readable program. You can help yourself by omitting all unnecessary spaces when typing the program in (including the spaces between the line number and the instruction that follows). This is also the reason why you will find reals rather than integers used throughout most of the program. Although integers use less space, every reference to an integer uses an extra byte (for the '%') which often outweighs any advantages.

IMPROVEMENTS

This month I want to describe a number of minor improvements to the original program. These both help to make the program more foolproof and more efficient, and pave the way for the next major additions. I have introduced three new utility functions called PROCread, PROCwrite and PROChead. The main purpose here is to separate the functions of:

read record from file
display record on screen
write record to file

The procedure PROChead simplifies the task of updating the screen information header. These new functions, the other lines that are changed as a result, and a number of other minor improvements are listed below. The best way to update the original program is to type in these lines and save as a *SPOOL file which can then be *EXECed with the original program. You will also need to delete lines 6100 and 6120 from the original program.

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```
 1050 delete%=0
 1060 DIM com$(20),record$(maxf,1),field$(maxf),width%(maxf),
      fp%(2),os 40
 3630 IF rec>recn THEN PRINT"File full":ENDPROC
 3660 PROChead(64,0,"Record: ",rec,4)
 3680 FOR I=1 TO f:record$(I,0)=FNinput(13,L*(I-1)+1,width%(I),"."):
      NEXT I
 3700 PROCwindow2:IF FNask("Confirm (Y/N): ")<3 THEN
      PROCwrite(rec,0):rec=rec+1:PTR#F=0:PRINT#F,rec
 3720 PROCwindow1:PROChead(47,0,"",rec-1,4):PROCwindow2
 4960 PROCread(I,0):PROCdisplay1(I)
 5240 PROChead(64,0,"Record: ",n,4)
 5280 FOR I=1 TO f:PRINTTAB(13,L*(I-1)+1)record$(I,0):NEXT I
 5620 PROCread(n,0):PROCdisplay1(n):PROCwindow2
 5660 IF A<3 THEN FOR I=1 TO f:record$(I,0)=STRING$(width%(I),"."):
      NEXT I:PROCwrite(n,0):delete%=-1
 6030 IF rec=1 OR NOT delete% THEN 6160
 6080 PROCread(I,0):IF ASC(record$(1,0))<>46 THEN J=J+1:
      IF I<>J-1 PROCwrite(J-1,0)
20700 DEF PROCread(n,R):LOCAL I
20720 PTR#F=256+recs*(n-1)
20740 FOR I=1 TO f:INPUT#F,record$(I,R):NEXT I
20760 ENDPROC
20800 DEF PROCwrite(n,R):LOCAL I
20820 PTR#F=256+recs*(n-1)
20840 FOR I=1 TO f:PRINT#F,record$(I,R):NEXT I
20860 ENDPROC
21000 DEF PROChead(x,y,msg$,n,w)
21020 PROCinv(1):PRINTTAB(x,y)msg$;
21040 PRINT SPC(w-LEN(STR$(n)));n
21060 PROCinv(0):ENDPROC
```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

ADDING TO THE PROGRAM

This month we are going to add two new features to the program involving three new commands and a modification of an existing command. These are:

COMMANDS
CREATE <filename>/<n>
FORMAT <filename>
PRINT <n>

The first of these is quite simple and allows a list of all recognised commands to be displayed on the screen. The procedure will automatically cope with any additional commands.

The other three commands are concerned with printing (and displaying) formatted records. Any number of formats can be set up for use with a data file. Each format (or print form) is a special one-record file created with a variation of the CREATE command as above. The value of 'n' specifies the number of lines that the form is to contain (e.g. CREATE form1/5). Once the format file has been created, the format record is generated by using the standard OPEN and ADD commands as for any datafile. You will find that the format record consists of a 'header' line followed by the number of lines specified at create time.

The header line should contain three numbers separated by commas, the number of forms per page, the width of a form (in characters), and the gap or number of lines between forms. Each of the form lines that follow should contain any, or none, of the following elements, in any order, separated by commas:

<n>	number of spaces
@<n>	at position n (i.e. tab n)
"<text>"	string or text literal
<text>	fieldname

The contents of a line will normally be left justified on the form but may be right justified or centred by appending /R or /C respectively to the end of a format line specification.

For example, a form for printing address labels might be specified as:

Header	8,30,2
Line1	"The Managing Director"/C
Line2	ADDRESS1
Line3	@4,ADDRESS2
Line4	@8,ADDRESS3
Line5	POSTCODE/R

This would print 8 forms per page consisting of 5 lines of 30 characters, and with 2 blank lines between forms.

Of the other two new commands, FORMAT is used to select a previously defined format, while PRINT will output records according to the format selected. As with the DISPLAY command, PRINT can be used to print all records or a single specified record, useful for checking a format for correctness. If you haven't a printer immediately available enter *FX5,0 to see the effect of the PRINT command. The printer output then just disappears into a 'sink' (see User Guide page 423).

The additional and amended lines for these enhancements are listed below and may be added to the original program as described for the earlier improvements, which are an essential pre-requisite.

Next month we will look in more detail at the functions and procedures used to implement the FORMAT and PRINT commands. We will also add further functions to the BEEBUG Filer including a much needed edit record facility. For those interested in programming there is, following the additional lines listed below, a more detailed technical description of the whole program which explains some of the basic ideas and describes how the program has been implemented.

****

NEVILLE'S NOTE: Pay particular attention to the difference between '1' and 'l' in the lines 7000 and on.

```
 1040 maxf=12:X=0:Y=0:format%=0:*FX4,2
 1100 DATA 10
 1160 DATA DELETE,END,COMMANDS,FORMAT,PRINT
 1670 IF C=5 THEN PROCdisplay(pm$,0)
 1700 IF C=8 THEN PROClistc
 1710 IF C=9 THEN PROCformat(pm$)
 1720 IF C=10 THEN PROCdisplay(pm$,1)
 2550 I=INSTR(p$,"/"):IF I THEN PROCf1(I):GOTO 2820
 3000 DEF PROCf1(i):LOCAL I
 3020 f=VAL(MID$(p$,i+1))+1:IF f>maxf THEN PRINT"Too many lines":
      ENDPROC
 3040 recn=1:p$=LEFT$(p$,i-1)
 3060 FOR I=1 TO f:field$(I)="Line"+STR$(I-1):width%(I)=64:NEXT I
 3080 field$(1)="Header":ENDPROC
 3100 :
 3320 FOR I=1 TO f:INPUT#F,field$,width%(I):
      field$(I)=FNstrip(field$,"."):NEXT I
 4800 DEF PROCdisplay(p$,flag)
 4890 IF flag AND NOT format% THEN PRINT"No format":ENDPROC
 4900 PROCwindow1:IF flag=0 PROCrecord
 4910 IF flag VDU28,0,19,79,4,12,2:form=0
 4960 PROCread(I,0):IF flag THEN PROCprint1(I) ELSE PROCdisplay1(I)
 4980 IF I<end AND flag<>1 THEN G=GET
 5000 NEXT I:IF flag VDU12,3
 5020 PROCwindow2:ENDPROC
 6300 DEF PROClistc:LOCAL I
 6320 PROCwindow1:CLS
 6340 FOR I=1 TO N:PRINT com$(I):NEXT I
 6360 PROCwindow2:PRINT"Press any key to continue";
 6380 I=GET:PROCheader:PRINT:ENDPROC
 6400 :
```

```
 6500 DEF PROCformat(p$):LOCAL F,I
 6520 IF p$="" THEN PRINT"No file given":ENDPROC
 6540 F=OPENUP(p$):IF F=0 THEN PRINT"No such file":ENDPROC
 6560 INPUT#F,frec,I,I,ff:IF frec=1 THEN PRINT"No format specified":
      ENDPROC
 6580 PRINT"Loading format ";p$
 6600 PTR#F=256:FOR I=0 TO ff-1:INPUT#F,p$:
      record$(I,1)=FNstrip(p$,"."):NEXT I
 6620 p$=record$(0,1)+",":FOR I=0 TO 2:p=INSTR(p$,","):
      fp%(I)=VAL(LEFT$(p$,p-1)):p$=MID$(p$,p+1):NEXT I
 6640 CLOSE#F:format%=-1:ENDPROC
 6660 :
 7000 DEF PROCprintl(n):LOCAL I,p,p$,l$,s$
 7020 FOR I=1 TO ff-1
 7040 l$="":s$="":p$=record$(I,1):IF p$="" THEN p=0:GOTO 7180
 7060 p=INSTR(p$,"/"):IF p s$=RIGHT$(p$,1):p$=LEFT$(p$,p-1)
 7080 p$=p$+",":REPEAT:p=INSTR(p$,",")
 7100 IF ASCp$=34 THEN l$=l$+MID$(p$,2,p-3)
      ELSE IF ASCp$>48 AND ASCp$<58
      THEN l$=l$+STRING$(VAL(LEFT$(p$,p-1))," ")
      ELSE IF ASCp$=64
      THEN l$=l$+STRING$(ABS(VAL(MID$(p$,2,p-1))-LENl$)," ")
      ELSE l$=l$+FNstrip(FNfield(LEFT$(p$,p-1),0),".")
 7120 p$=MID$(p$,p+1):UNTIL p$=""
 7140 l$=LEFT$(l$,fp%(1))
 7160 p=fp%(1)-LENl$:IF s$="C" THEN p=p/2 ELSE IF s$="R" ELSE p=0
 7180 PRINT SPCp;l$
 7200 NEXT I:PROCline(fp%(2)):form=form+1
 7220 IF form>=fp%(0) VDU12:form=0
 7240 ENDPROC
 7260 :
21200 DEF FNstrip(p$,c$):LOCAL I:I=LENp$+1
21220 REPEAT:I=I-1:UNTIL MID$(p$,I,1)<>c$
21240 =LEFT$(p$,I)
21260 :
21400 DEF PROCline(n):LOCAL I
21420 IF n>0 FOR I=1 TO n:PRINT:NEXT I
21440 ENDPROC
21460 :
21600 DEF FNfield(p$,R):LOCAL I:I=0
21620 REPEAT:I=I+1:UNTIL p$=field$(I) OR I>f
21640 =record$(I,R)
21660 :
31010 IF ERR=17 THEN PROCheader:PRINT'"Command aborted":GOTO 140
```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

THE FILE STRUCTURE DESCRIBED

A database file will consist of a series of fixed length records, and each record will contain a number of fixed length fields. Field and record lengths need to be fixed in any given data file so that they can be accessed using the PTR# instruction. We also want our program to be able to handle a variety of different data files, each with its own field and record lengths.

One solution to this problem is to reserve space at the start of any file in which to store a description of that file. By always keeping the file description in the same format, the one program can readily handle many different files.

BEEBUG Filer reserves the first 256 bytes in any file for the file description. This is then followed by the records themselves as shown in figure 1.

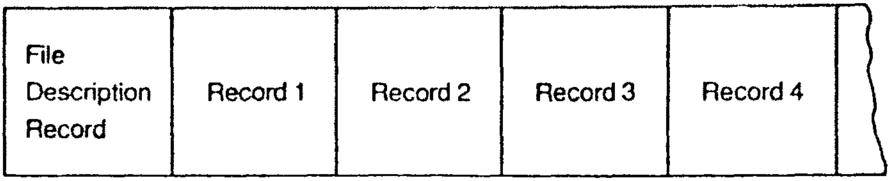


Figure 1

The file description contains the position of the next free record in the file (or, if you like, the number of records in use plus 1), the maximum number of records that the file can hold, the size in bytes of each record, the number of fields in each record and the name and length of each field in turn. The structure of the file description record (FDR) is shown in figure 2. Field names, the only variable length items, are limited to 12 characters (14 characters of file space) and the 256 byte FDR provides enough space for up to 12 fields, which is therefore the limit for any BEEBUG Filer database file.

You could increase this by allocating, say, 512 bytes for the FDR but there are other limitations too. The area of the screen used to display a record has a maximum of 17 lines available. This would be the maximum number of fields that could be displayed together. If you want to change this or experiment, then you will need to alter the value of 'maxf' in line 1040 to whatever limit you wish to set, and change 256 to 512 (or whatever) where referenced with PTR#.

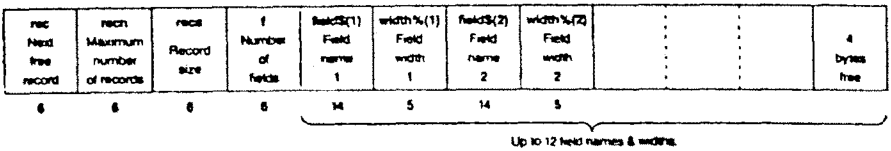


Figure 2

COMMAND MECHANISM

BEEBUG Filer is command-driven, with the number of commands and the commands themselves read from DATA statements in PROCsetup at lines 1080 to 1160. These are read into an array for subsequent use. This makes the addition of further commands very easy, as described earlier.

The program operates a simple loop, repeatedly calling the main command procedure (PROCcommand). This procedure, at lines 1500 to 1880, inputs the next command from the keyboard, checks its validity, and depending upon the command entered, calls one of the several key procedures.

KEY PROCEDURES AND UTILITY PROCEDURES

The key procedures implemented in the basic program are as follows:

PROCcreate - create a new datafile
PROCopen - open a datafile for use
PROCadd - add a record to the file
PROCdisplay - display one or more records on the screen
PROCdelete - delete a specified record
PROCclose - close a data file

These key procedures are listed from line 2500 upwards and include a number of supporting procedures and functions. All the key procedures contain a number of checks, displaying appropriate error messages as needed.

There are also several general utility functions and these are listed from line 20000 upwards:

FNask - seek a Y/N answer
PROCheader - display main header
PROCwindow1 - set record display area
PROCwindow2 - set command display area
PROCinv - toggle inverse video effect

GLOBAL VARIABLES

A number of variables are used throughout the program and these are listed below. All variables that are logically local to any procedure or function are declared as LOCAL to avoid any unforeseen clashes.

Global	Function
rec	Next free record
recn	Maximum number of records
recs	Size of a record in bytes
f	Number of fields

Global	Function
field$()	Fieldnames
width%()	Fieldwidths
record$()	Contents of fields for any record
com$()	List of commands
maxf	Maximum number of fields
open%	Flags if a datafile open
F	Datafile channel number
F$	Current datafile name
exit%	Flags if exit from program
N	Number of commands in use
C	Number of current command
X	Used to record the position
Y	of the cursor in window 2
L	Set to indicate single or double line spacing depending on number of fields
delete%	Flags if any record logically deleted

More technical information will appear next month covering the later additions.

****

CONTEST

REMINDER:

Don't forget the CONTEST initially mentioned in the August 1986 Beeblet!

Entries close at the end of November 1986.

For those who can't find their August Beeblet, the competion is for a games program played interactively between two Acorn machines connected together via a serial link - ie., RS232 or even the cassette port.

HINT: Make sure that clear instructions are either included in the program or are supplied seperately.

The prize? One box of 5.25" floppy discs or equivalent.

Don't forget - to supply your name, address and telephone number with your entry.

Entries close at the end of November 1986.

Hurry, This is the last reminder!

****

E2P SECOND PROCESSOR

Ken Fredericksen of Nelson has written to the User Group with information about the E2P Second Processor for the Electron. He is obviously fairly wrapped with the end result, even though it took him some 4 - 5 hours to fully assemble one board. Copies of the information supplied by Ken can be made available at a cost of $3 to cover the cost of copying and postage (enquiries to Electron Second Processor, BBC/Acorn Computer User Group, PO Box 9592,Wellington).

All the ICs are available in New Zealand, which along with reistors and capacitors would cost about $100. He has supplied masks for the printed circuit board, and arrangements could be made to make and supply the boards if a firm requirement is advised.

Ken says he uses the second processor all the time now except when running the Acornsoft Database program - which locks up for some reason. Comments about other programs are:

View	excellent - 30K available in any mode; faster
Viewsheet	excellent
Starfinder	runs faster, and can now print the star maps.

He is interested in any feedback from others who may have/or are interested in the second processor.

****

On Creatively Using Shadows...

Part 2

Steve Cosgrove

The three OSBYTE commands dealing with shadow memory (*fx108, *fx112, *fx113) are briefly documented in the Welcome Guide. I have added these notes from my own experiments. None of these calls has been used by Acorn before, as far as I know. Each operates "independantly" so to speak - so useful / interesting effects can be generated by some combinations. I will comment on some of these after each description.

A = &6C (*fx108) Switch main / shadow memory into main map: X = 0 (*fx108,0) switches main memory into main map X = 1 (*fx108,1) switches shadow memory into main map.

This command affects all the memory from &3000 to &8000. *fx108,1 pages out ("moves to the right" in diagram) the main memory, and pages in ("moves to the left") the shadow memory. This will be all the screen memory in MODEs 0, 1, & 2. The command will affect any program or data in memory above &3000. If PAGE is set to &3000 before loading a program, the use of this call removes that program totally from the main memory. Another program could then be loaded at the same value of PAGE. The second program would live in "shadow memory", but would RUN quite happily until *fx108,0, which would cause it to disappear into the shadows again. At this stage RUN would cause the first program to run as if nothing had happened in the meantime.

A disaster can be caused by *fx108,1 if it is made by a program when HIMEM is higher than &3000. Any language is free to use all the space between PAGE and HIMEM. *fx108,1 can effectivly steal part of the program, and/or it's variables' values, while the program is trying to use them! The problem can, however, be avoided if precautions are taken.

A small program presents no problem at all. If HIMEM is set to &3000, (default in MODEs 0-2) then there will be no effect on the running of the program at all. In fact I can't think of any good use for *fx108 in this situation.

The most obvious use for *fx108 is in a larger program which requires graphics. Such a program may not fit into the space left after a screen mode is set. It will run, however, when shadow memory is used for the screen. During the running of the program, it may be desirable to "save" a screen to disk or tape using *SAVE filename 3000 8000. Normally this method is not posible because *SAVE will only work on main memory. The answer is to page shadow memory into the main map, then save the screen, and immediately page the main memory back -

```
DEF PROCsave_screen
*fx108,1
REM switch in shadow RAM. NB, on no account
REM    can any variables be changed now!
*save Screen_img 3000 7FFF
REM save screen. Note long filename (ADFS!)
*fx108,0
REM switch in main RAM. Program in now intact
REM    again.
ENDPROC
```

The PROCedure which uses *fx108,1 should be near the begining of the program. So long as the program does not use GOTOs, it should be no problem to move one PROCedure in this way. It is not necessary that it be very near the begining. The following will determine whether it is near enough:

```
SAVE the program
DELETE all lines after the ENDPROC of save_screen
PRINT ~TOP
note the value given
reLOAD the program again before you forget to.
```

If TOP is less than &3000, then the program should work fine. If it does not, then the program will crash with unpredictable results, so be careful to SAVE it before a RUN, just in case. If the program should crash, remember to reset the memory (*fx108,0 or hit BREAK) before LOADing it again, or the computer will get very unhappy.

If it is necessary to use a machine code program such as a screen-print which access the screen memory directly, then similar rules apply. Space cannot be set aside for the routine by using the DIM statement. An alternative would be as follows:

> supposing the routine needs less than &100 bytes in which to run, and this used to be created using 'DIM code% &100' before loading the program (perhaps in a header program)

```
C% = PAGE
PAGE = PAGE + &100
CHAIN "Graphics_1"
```

within Graphics_1, replace DIM code% &100 by:

```
code% = C%
:
REM start of space reserved in
REM header program for
REM machine code.
```

In this case the following should be added to the machine code program before any memory is read:

```
OSBYTE = &FFF4
:
REM this is in the BASIC part
[
...
 LDA #&108  /OSBYTE call to switch
 LDX #&1    /the screen into main
 JSR OSBYTE /memory map
...
...
/main screendump comes here
...
 LDA #&108  /OSBYTE call to switch
 LDX #&0    /the main memory back into
 JSR OSBYTE /the memory map
...
RTS
]
```

Using the procedure to set up the machine code will mean that it can be called (using 'CALL code%') from anywhere within the BASIC code.

****

A TV OF MANY COLOURS

by Paul Collins

Grumbles again from the kids about the BEEB tying up the family colour TV? Getting sick of mode 2 screens in glorious black and white? What you need is a small colour TV with audio and video inputs; something like the ones advertised in the User magazines - if only they were available here!

This scenario, no doubt, has occurred many times to most of us (while spilling coffee and breadcrumbs over the latest mag.) making one wish one was richer / lived in the UK (Ed: you've got to be kidding) / collected stamps. Well perhaps you may be interested in a portable TV that Philips have on the market here in En Zed.

Briefly, it's specifications are:

Model	14CF1114/79S
Screen	36cm (14")
W*H*D	41*32*35 cm
Weight	OK - thats not metric!!
Sockets	Audio in / out RCA Video in / out RCA Headphones 6.5mm stereo compatible RGB (yes Red Green Blue) TTL 6 pin Din Audio in also on RGB socket.

And how does it perform? In only two days, I have found it to be just what I wanted, the only concern being the so-so definition on 80 column text (see comments later). As a portable TV set it does all the right things, without any frame buzz, sensitivity on the supplied rabbits ears about state of the art, and once set up is easy to use, having only volume, power and channel controls on the front panel.

Setting up requires the opening of a flap below the screen, giving access to the brightness, contrast, colour and 10 channel preset controls. Adjusting these controls does require some deftness, especially holding on to the infernally tiny screwdriver used for tuning - but as they are usually only set once, I guess that's acceptable. You can't forget to put the screwdriver away in it's hole, as the green tuning line stays on screen while it's out.

On the BEEB the results varied depending on screen mode and input to the TV used. Using the antennae input gave the poorest performance, naturally, the colour requiring a bit of boost, and 80 column text was almost unreadable. Direct video from the BEEB's BNC socket was clear, but, of course, in black and white. The RGB socket gave the best performance, with clear, bright colours and readable text in mode 3. However a proper green screen is still the better proposition if you are going to do much text work in 80 columns, as the text lacked the sharpness of a monochrome monitor.

What is RGB, you ask. Essentially they are the primary colours from which all other colours are made. At the TV station, these colours are combined with several other bits of information and added to the station carrier wave for your TV to receive. In your TV, the reverse occurs, the other bits are stripped off and the Red, Green and Blue signals are sent to the picture tube. RGB signals are available directly at the back of the BEEB, allowing us to connect directly into the TV's RGB guns on the picture tube, so we can avoid the inevitable distortion introduced during the process of adding and subtraction of signals. One small point about the RGB standards is that there are two: ie., TTL and Analogue, but for once we win - the BEEB and this TV are compatible with both using TTL.

The 36 page manual had a whole 4 pages in English, but didn't have the RGB pin connections in it. Fortunately these are printed on the case around the socket. The plug required is NOT included and may be difficult to obtain - although if you dont use the audio input, a 240 degree, 5 pin Din will do and allows you to use quad audio cable. The diagram below show the pins from the back of each plug.

Dollars is what it is all comes down to, of course - the recommended retail price (incl GST) is $950. However, if you shop around............... (Mine was $890)

……

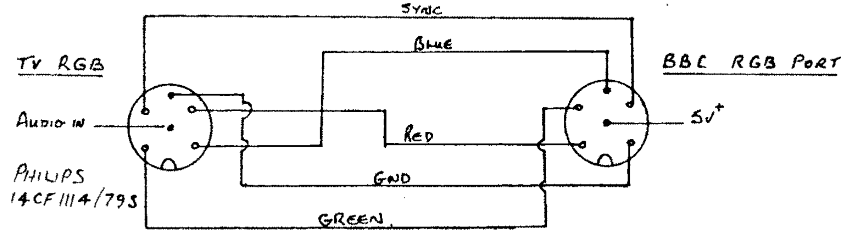


FIGURE #1 A TV OF MANY COLOURS (FROM THE BACK OF THE PLUGS)

* * *

FOR SALE

BBC Model B Computer - Serial No. 01-ANB03-5030753
Approximately two years old. Includes
'VIEW' (Wordprocessor) and 'VIEWSHEET' (Spreadshhet) ROMS

Plus Double Sided Double Density 80 Track Disc Drive No 4132 (360K)

Plus Manuals etc., as follows:

- User Guide manual for the BBC
- VIEW User Guide
- VIEWSHEET User Guide
- BBC Basic for Beginners
- Disc System User Guide
- Computer Study and Revision
- BBC Micro Data Log
- Computer Games
- Electron Machine Code for Beginners
- BBC Micro - Beginners Micro Guides
- Miscellaneous BBC Documents

Plus Twelve 5.25" Floppy Discs with a large selection of games of all types, plus Demo Disc, Utilities etc.

Price: $1,800

Contact: Tony Loughlan, telephone (070) 775-850
24 Franklin Terrace
Havelock North

NEWS NEWS NEWS NEWS NEWS

The Nelson User Group has forwarded the following item published in the *Nelson Evening Mail* during August which may be of interest to *Beeblet* readers:

> A $45,000 computer investment commissioned this week puts Nelson College among the leaders in secondary school education.
>
> Nelson College was one of the first to install a computer when Dave Nightingale commissioned an Alpha donated by the McKee Trust back in 1974.
>
> This week 15 BBC Master 128K computers, networked to a BBC Master Turbo, were installed complete with colour screens, word processor, and spreadsheet in memory.
>
> They compliment the two BBC stand-alone computers and the five Apple computers already in use.
>
> Students' interest in the science has increased every passing year, said Dr Keith Boardman, Head of Computer Studies since 1975.
>
> Today almost all third form students are introduced to computers with classes on their history, their uses and basic programming.
>
> The number interested decreases as the students continue through school, and by the time they reach the sixth form there are about 40, he said. From college, many continue on the Central Institute of Technology or to University for degress in Computer Science, Dr Boardman said.
>
> Funding for the new computers and relining of the computer room is being met by equal shares by the board of Governors, the Parents Association and the students.

The item was illustrated with sixth form students operating the new BBC Master computers - unfortunately, our printing processes do not permit its duplication.

....

Another item from Nelson:

Some members fitted capacitors in the audio circuits of their BEEBs to increase the volume - it worked, but introduced a new facility: their computers were permanently tuned to Radio Nelson (1341kHz).

::

Please forward any items of news relating to BBC/Acorn computers that you think might be of general interest to readers of *Beeblat.*

Address any items to:

BEEBLET
BBC/ACORN Computer User Group of NZ, Inc
PO Box 9592
WELLINGTON

We know how he feels!

HELP

A Forum Conducted by Steven Rose

Only one problem received this month - Mitch Park of Hastings enquires:

How do I use *COMPACT on the Master's ADFS? What do the parameters <SP> and <LP> mean?

This is an example of the benefits of Reference Manual Part One.

The syntax is: *COMPACT [<start page number> <number of pages>]

<start page number>	is a two digit hexadecimal value indicating the start address for the buffer used by the compact routine.
<number of pages>	is the number of pages to be used, also a two digit hexadecimal value.

If you use a non-shadow mode the parameters are optional, but if you leave them out you'll get all sorts of weird screen effects, which don't matter as long as you don't mind the screen contents being lost (Sideways ROMS sometimes make interesting reading in Mode 7 !).

With shadow modes, the parameters are mandatory.

The manual says "Care must be taken to ensure that the absolute address of the end of the buffer is in the IO processor's RAM (ie < &8000)".

It also says that the speed of *COMPACT depends upon the size of the buffer, and recommends using a shadow mode and the command:

```
*COMPACT 0E 72
```

This uses the entire user RAM in the IO processor.

Finally, <SP> and <LP> are <Start Page> and <Last Page> respectively.

With no other problems received this month, I'll conclude with:

Comments 'N Hints

I have grouped these into two categories, viz, BBC Masters and Programming-related:

BBC Masters

1. My opinion of the Master, for what it's worth, is that they seem to be worth having if you were thinking of upgrading your BBC anyway (perhaps some shadow screen RAM, or sideways RAM), or if you want to do fancy graphics, e.g. animation, or faster Mandelbrot pictures. The former is the main reason why I bought one and it seems worth it, apart from having various unplayable games, like ELITE, Castle Quest, etc. At least fifty percent of my games do seem to work though, so life's not TOO bad.

2. I want to emphasise that only those machine code programs written for a 65C02 or 65C12, e.g. BASIC (which is a machine code program in ROM), will run faster. REVS for instance seems to run at exactly the same speed as on a BBC.

Those programs, like the BASIC ROM, which HAVE been so written run faster because they use less instructions to do the same thing.

This means that a BASIC program takes less machine code instructions to execute on a MASTER than on a standard BBC, so that each statement will take less time on the MASTER, although both machines run at the same clock rate (i.e. a "load accumulator" machine code instruction on a BBC takes the same time as a "load accumulator" instruction on a MASTER).

This is also why there are more facilities in BASIC on the MASTER: less instructions to do the same tasks means some room left over, so you may as well add some extra features.

Programming

1. A left square bracket followed by Return (i.e. [<RETURN>) is a valid BASIC command, and returns the current value of P% in hex.

2. CLOSE#0 closes ALL files, including *EXEC files that are executing! This can be used to stop the execution of an *EXEC file part way through it.

3. The number of remaining characters in the printer buffer can be checked with ADVAL(-4) or ADVAL(&FC). Therefore, to see if the printer is turned on and on-line, see if ADVAL(-4) is less than 63, after sending it a character. The following sends a "delete current line" character to the printer, waits for a tenth of a second, and then checks to see if the character is still in the buffer (if so, then the printer isn't talking to the computer):

```
VDU 2,21,3:Z%=INKEY(10):IF ADVAL(-4) < 63 THEN PRINT "Printer
is not ready"
```

****

HALL OF FAME

PUBLISHER:	GAME	HISCORE	SCORER
Aardvark	Frak		
	Monsters		
	Zalaga		
Acorn	Arcadians		
	Aviator		
	Countdown		
	Meteors		
	Monsters		
	Planetoid		
	Rocket Raid		
	Snapper		
	Starship Command		
	Super Invader		
A&F	Cylon Attack		
	Chuckie Egg		
Alligat.	Bug Blaster		
Amcom	Fortress		
Bugbyte	Twin Kingdom		
Comp.Con	Android Attack		
Dr Soft	'747'		
Firebird	Bird Strike		
FG Soft.	Mutant Defender		
	Lords of Time		
	Snowball		
Micro Pow	Castlequest		
Prog Pow	Felix in Factory		
	Frenzy		
	Killer Gorilla		
	Laser Command		
	Moon Raider		
Sup.Soft	Q*Bert		
	Mr Wiz		
	Roadrunner		
?	Dr Who		

*Electron Version.

HALL OF FAME

Editor's Note:

It appears as though all the high scores recorded to date have slipped down the side of the BREAK key. What a marvellous opportunity for some newcomers to record their high scores. The BREAK key gremlin has told me that he(she?) will be swallowing any listing that is over six months old - but I might just be able to persuade him(her) to reveal the old record list should anyone manage to exceed it. There is, of course, no time limit on holding the overall record.

Have a go!

GAMES MENU LABEL PRINTER

by J.P.Colvin

WELL!. I vowed this was never going to happen! Whenever someone asked for contributions to Beeblet at a clubnight, I got a case of terminal lint on my trousers (Sneaky side-long glances showed lots of others with the same problem). Besides, what did I know, when compared to the hotshots I listened to in reverent awe?

Apparently I didn't hide well enough, so here we are!

Over the last couple of years, I have noticed quite a number of members using a menu program written by Robert Ngan, for their games discs. This program uses DATA statements to hold the full name of the game, along with the file type, filename and disc side.

The advantage of this menu is that the name of the game is displayed in full, and I feel this more than offsets the time taken to make the DATA entries. I use this menu on all my games discs.

Although DISCDEX and similar programs, offer the facility to print a label for discs, these only duplicate the catalogue entries with a 7 character maximum. It seemed a pity to have the full names of games in DATA statements and not use them on disk labels, so I added a label printer option to Robert's menu. Using The BASIC Editor's APPEND command helped to transfer all the DATA lines from the original menu program to the amended version, but this was no small task.

While completing this chore, I saw the light. All the information I needed was there in the original menu files, and the full name of the games was there preceeded by the word 'DATA'. I only had to write a program that worked through each menu, byte by byte as a sequential file, looking for the BASIC 'DATA' token (&DC), and having found it, read the name of the game following it into an array for formatting and printing.

The following program is the result. As it stands it is set up for a Logitec FT-5001/FT-5002, Super 5, or Panasonic 1091, but should run on most Epson compatible printers. If your printer does not have a NLQ mode delete line 930. The printer commands section is deliberately kept to one command per line with an explanatory 'REM', to simplify changes. The labels I use are quik-stik MR 36-103 (R23-65), which measure 104 x 36mm and are sold in rolls of 100 at Whitcoulls. They were chosen as they are the largest I could find that will fit on a disc, and still show above the sleeve.

DESCRIPTION

110 sets default menu name to 'MENU'. If the menu name is not 'MENU', this is trapped at 760 and the menu name is requested.
120 maximum of 36 games is catered for.
130 opens the menu file as a sequential file.
140 steps 1300 bytes through the file to save time searching for 'DATA' tokens that are at the end of the menu anyway.

160-200 repeats:-

PROCfinddata: looks for a byte containing &DC (BASIC 'DATA' token), and uses PROCnewline to step past line numbers which could contain erroneous &DC bytes.

PROCmakename: having found a byte containing &DC reads the name of the game byte by byte until a comma is encountered, denoting the end of the name. until: 35 names are assembled, or the End of File is reached.

PROClabel: prints a message on how to line up the label, and would need to be altered for different printers. You are prompted for a title for the label and a disc number (which will be ignored if <RETURN> is pressed). It then calculates which of 2 font sizes to use, and adjusts the linespacing for best effect.

PROCprinter: prints to the screen only, the title and names as they will be used on the label.

PROCconfirm: asks permission to proceed if satisfied, or abort.

250 VDU2:PROCprinter: turns on the printer, and repeats PROCprinter to produce the label.

Even if this program isn't of direct use to you, the general idea of using one program to 'loot' information from another may tickle a brain cell somewhere.

If you would like a copy of this program, see me at a clubnight or send a disc/tape, along with a stamped, self-addressed envelope to:

J.P.Colvin,
600 High Street,
Lower Hutt.

Or, better still, request a copy from the User Group on your disc or tape when it is returned after you have submitted your own favourite program, article and/or review for *Beeblet*.

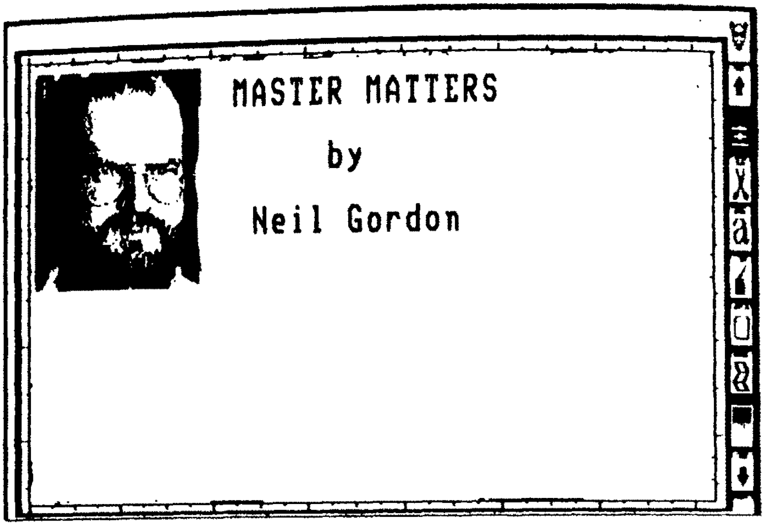
::

```
 10 REM              *** GAMES DISC LABEL PRINTER ***
 20 REM                   by John Colvin,  Sep. 86
 30 :
 40 REM  For use with Games Menu program written by Robert Ngan.
 50 REM  Prints a disc label, giving the names of games in full.
 60 REM  (may work with other menu programs  which hold names of
 70 REM             the games in 'DATA' statements.)
 80 :
 90 MODE3
100 ON ERROR GOTO 760
110 progname$="MENU"
120 DIM name$(36)
130 X=OPENIN(progname$)
140 PTR#X=1300
150 X%=1
160 REPEAT
170 PROCfinddata
180 PROCmakename
190 X%=X%+1
200 UNTIL X%>35 OR EOF#X
```

```
210 MODE3
220 PROClabel
230 PROCprinter
240 PROCconfirm
250 VDU2:PROCprinter
260 END
270 :
280 DEFPROCgetchr
290 IF NOT EOF#X THEN CHR%=BGET#X
300 ENDPROC
310 :
320 DEFPROCfinddata
330 REPEAT
340 PROCgetchr
350 IF CHR%=&0D PROCnewline
360 UNTIL CHR%=&DC OR EOF#X
370 ENDPROC
380 :
390 DEFPROCmakename
400 REPEAT
410 PROCgetchr
420 IF CHR%<>44 THEN name$(X%)=name$(X%)+CHR$(CHR%)
430 UNTIL CHR%=44 OR EOF#X
440 ENDPROC
450 :
460 DEFPROCnewline
470 FOR J%=1TO4
480 PROCgetchr
490 NEXT
500 ENDPROC
510 :
520 DEFPROClabel
530 PRINTTAB(5,0)"To print a label on a  LOGITEC FT-5002 or
    PANASONIC 1091 printer,"
540 PRINTTAB(5,1)"using 'QUIK-STIK' MR 36-103 (R 23-65) labels,
    measuring 104 X 36mm"
550 PRINTTAB(17,2)"(Sold by Whitcoulls in rolls of 100.)"
560 PRINTTAB(12,5)"Align left side of label with '0' mark on
    tear-bar"
570 PRINTTAB(35,7)"THEN"
580 PRINTTAB(12,9)"Wind the label down until TOP edge is just
    visible"
590 PRINTTAB(12,10)"above the silver presser bar below the
    print-head"
600 INPUTTAB(20,19)"Enter the TITLE for your label "title$
610 leftspaces%=(30-LEN(title$))/2
620 rightspaces%=30-(leftspaces%+LEN(title$))
630 title$=STRING$(leftspaces%," ")+title$+
    STRING$(rightspaces%," ")
640 INPUTTAB(20,21)"Enter the disc number "dnumber$
650 topunderlines%=(LEN(title$)+LEN(dnumber$)+3)
660 linespace%=18+72/X%
670 IF X% <= 21 THEN linespace%=30+72/X%
680 leftmargin%=2
690 windup%=75:                   REM The distance the label must
                                      be moved up from the
                                      alignment point to the start
                                      of printing point. (In
                                      inches/216)
```

```
 700 ENDPROC
 710 DEFPROCconfirm
 720 PRINT'''TAB(8)"If this is correct, press 'P' to print a
     label,"''TAB(18)" or 'SPACE' to restart";
 730 G$=GET$
 740 IF INSTR("Pp",G$)= FALSE THEN RUN
 750 ENDPROC
 760 IF ERR<>222 THEN REPORT:PRINT" at line number ";ERL:END
 770 VDU26,12:                    REM Reset text window and CLS
 780 VDU28,20,24,79,0
 790 *CAT
 800 VDU28,0,24,79,22
 810 PRINTTAB(16)"There is no file named 'MENU' on this
     disc."''TAB(14)"Please enter the filename of your menu
     program ";
 820 INPUT progname$
 830 GOTO130
 840 END
 850 DEFPROCprinter
 860 CLS
 870 VDU28,10,24,79,5:              REM Defines a text window to
                                        centre catalogue on screen
 880 VDU1,27,1,64:                  REM Resets printer
 890 VDU1,27,1,51,1,windup%:        REM Winds label up to starting
                                        point
 900 PRINT
 910 VDU1,27,1,64:                  REM Resets printer
 920 VDU1,27,1,108,1,leftmargin%:   REM Sets left margin
 930 VDU1,27,1,110:                 REM Selects NLQ mode.  (delete
                                        for non-NLQ  printer)
 940 VDU1,27,1,45,1,1:              REM Selects Underline mode
 950 VDU1,27,1,69:                  REM Selects Emphasis mode
 960 PRINTSTRING$(topunderlines%," ")
 970 PRINTtitle$;
 980 IF dnumber$<>"" PRINT" (";dnumber$;")" ELSE PRINT"    "
 990 VDU1,27,1,64:                  REM Resets printer
1000 VDU1,27,1,108,1,leftmargin%:   REM Sets left margin
1010 VDU1,27,1,51,1,linespace%:     REM Selects linespacing
1020 VDU1,15,1,27,1,71:             REM Selects printer
                                        condensed-double mode
1030 IF X% > 21 THEN VDU1,27,1,83,1,1: REM Selects subscript mode
                                           if more than 21 games on
                                           disc
1040 FOR I%=1 TO X%
1050 name$(I%)=LEFT$(name$(I%),20):    REM Reduces name to max of
                                           20 chrs
1060 REPEAT name$(I%)=name$(I%)+" ":UNTIL LEN(name$(I%))=22:
                                       REM make 22
1070 NEXT
1080 FOR I%=1 TO X%
1090 PRINTname$(I%);
1100 IF I% MOD 3 = 0 THEN PRINT
1110 NEXT
1120 PRINT
1130 VDU1,27,1,64:                     REM Resets printer
1140 VDU3
1150 ENDPROC
```

****



MASTER MATTERS

by

Neil Gordon

Since my last column, I have at last managed to get my hands on a Master Reference Manual Volume 1. It cost $65 (plus GST, now), and should be available through your dealer or else direct from Barsons. I highly recommend buying it - the goldmine of information in there is on a par with that for the Advanced User Guide for the BBC B. It includes complete information on all the *FX (OSBYTE) and OSWORD calls, using the sideways RAM, memory maps, hardware information, and complete details of the filing systems including the OSWORD calls to directly read and write to disk. There is no comprehensive description of the hardware - that's left for the Master Advanced Reference guide; rumour has it that in spite of its being advertised it has not yet been written.

The section on paged ROMS includes an interesting item on the low nibble (that's 4 bits, or half a byte - get it?) of location &8006 - the ROM type flag. For the BBC B, all that mattered was that bit 1 (the second to lowest order bit) should be 1. Now, that nibble can be used to determine the type of code stored in the ROM, to indicate what kind of processor the code is intended for. As we would expect, the code can be for a 65C12 (the Master and the Turbo coprocessor), a Z80, a 32016 (as used in the Master Scientific) or an 80186 (as used in the Master 512). Intriguingly, there is also provision for an 80286 (that's the processor used by the IBM PC-AT and its clones), and a 68000 (as used by the Amiga and Atari 520 and 10409T). Apparently Acorn have no plans for making a 68000 coprocessor themselves, but have wisely provided for its inclusion by third parties.

Because the entire nibble is now examined, rather than just bit 1, there is a new MOS error saying "I cannot run this code" which happens when an attempt is made to select a paged ROM as a language when the ROM type indicates it is designed for use with a different processor. You may encounter this with some ROM images if they didn't set the low nibble of &8006 to &2.

By now all of you who bought Masters through a NZ dealer should have been notified about the battery upgrade - if you haven't, contact your dealer urgently and get them to fit it. If, like me, you got your Master from England, you can either ask your dealer there or do it yourself. The latter is less hassle, and doesn't cost much. If you open up the Master, you will see a link marked PL8 with three prongs on it over on the left (by the space for the modem). A lead goes from there to the battery beside the cartridge slots. I removed the connector on PL8, and substituted another one (with the correct polarity!) connected to a plastic battery pack holding three 1.5V penlight (AA) cells (alkaline are recommended - I've put NiCads in for the moment, but have been told that it isn't good for them to have only a small current drawn off). You could use the existing connector on the prongs and cut or desolder the wires to the Lithium battery.

I promised last time to tell you how to automatically date-stamp BASIC programs when they are saved to disk. The technique is actually a little naughty, as it involves overwriting the start of the program just before it is saved. It makes use of the new Master SAVE command which can be followed by a string expression, rather than just a string enclosed in double quotes. If you say SAVE FNname, then the function defined by DEF FNname will be executed (with the side effect of writing the current date and time into the start of the program), then return a string giving the name to save the program under.

In the following example I have chosen to return a name (DEMO_DATE) by which the program will always be known. If, instead, you wanted to save successive versions of a program under development under different names, then you could increment a counter (Z% or something) each time the function was called and return names TEST1 ("TEST"+STR$(Z%)), TEST2, and so on.

```
   10REM Thu,11 Sep 1986.21:42:47
   20REM Remarks describing the program
   30REM and some more
   40REM The next line defines the function
   50DEF FNname:$(PAGE+6)=TIME$: ="DEMO_DATE"
   60:
   70REM And now we continue with the program proper
   80REM Note that we don't need to GOTO 100 at the top since
   90REM executing a single line function will have no effect
  100END
```

Note that it is critical that the first line of the program should have exactly the right number of characters on it. There should be no space between the line number and the REM, one space after the REM, then exactly the number of characters required for the date and time string. If not, you may corrupt the program when saving it using SAVE FNname.

I also promised last time to tell you about how to have a 640*512 MODE 0 screen on the Master. Normally, a MODE 0 screen (which needs 20 kbytes of memory) is 640 pixels across and 256 pixels high. But, providing you have interlace mode on, you will see on a good monitor that there are actually 512 lines displayed on the screen. Each line in memory is displayed twice every 25th of a second, as two screen lines, one just above the other. That's where the term 'interlace' comes from - the lines are interlaced on the screen. The refresh rate of fifty times a second gives less flicker than would a rate of twenty-five times a second, and the flicker is reduced even further if interlace is turned off so that each line is written twice in the same place, with small gaps between the two lines.

Now, suppose that we fool the computer by alternatively switching, every fiftieth of a second, between the main screen memory and the shadow screen memory. No longer need the two lines show the same information. In effect, by writing some information to main memory and some to shadow we can get 512 lines of different information on the screen.
Unfortunately, there are problems. The first is how to draw the appropriate information in each 'screen' so that diagonal lines and so on are shifted in the right direction. The second is that we have no control over which set of lines is the higher and which the lower of each pair, so that drawing the diagonal line may look really weird - a bit like

```
              *                            *
               *                          *
            *         rather than        *
             *                          *
          *                            *
           *                          *
```

Finally, because the pairs of lines no longer display the same information, flicker is increased.

The following program demonstrates the super-high resolution MODE 0 screen - it is fully commented to explain how it works. You will also see that bold text can be produced by writing it to both screens. This can be seen without the interlace mode on, since it will still be written twice to the same place and appear brighter.

That's all for now - remember, send those queries in.

……

```
 10 REM Thu,02 Oct 1986.20:11:42
 20 REM Event routine to demonstrate high resolution
 30 REM 640 by 512 mode 0 on BBC Master-128
 40 REM (c) Neil Gordon
 50 DEF FNname:$(PAGE+6)=TIME$: ="HIRES_DEMO"
 60 :
 70 REM Ensure interlace is on
 80 *TV 0,0
 90 :
100 REM Clear both shadow and main screen
110 MODE 128 : MODE 0
120 :
```

```
130 REM Assemble event routine which will handle top of screen
    event
140 :
150 OSBYTE=&FFF4
160 EVENTV=&220
170 DIM code 100
180 FOR PASS=0 TO 2 STEP 2
190 P%=code
200 [OPT PASS
210 \ Stack all registers
220 PHP:PHA:PHX:PHY
230 :
240 \ 'switch' will have been set to 1 if we should
250 \ reverse the order of displaying the screens
260 \ The following tricky bit of code decides whether we
270 \ should now switch which one is displayed
280 :
290 LDX switch
300 TXA
310 EOR switch
320 STA switch
330 TXA
340 EOR flag
350 EOR #1
360 STA flag
370 TAX
380 INX
390 :
400 \ Use routine to set displayed screen, then
410 \ restore registers and exit
420 LDA #113
430 JSR OSBYTE
440 PLY:PLX:PLA:PLP
450 RTS
460 :
470 .flag:   EQUB 1   \ keeps note of which screen to display
480 .switch: EQUB 0   \ can be set to 1 to reverse order
490 :
500 ]
510 NEXT PASS
520 :
530 REM Point interrupt vector to our code
540 EVENTV?0=code MOD 256
550 EVENTV?1=code DIV 256
560 :
570 REM Enable events
580 *FX 14,4
590 : 600 REM Draw a box...
610 PROCdraw(100,100,800,100)
620 PROCdraw(800,100,800,800)
630 PROCdraw(800,800,100,800)
640 PROCdraw(100,800,100,100)
650 PROCdraw(100,100,800,800)
660 PROCdraw(100,800,800,100)
670 :
 680 REM Demonstrate bold text (drawn in both screens)
 690 REM versus not bold (drawn in one screen only)
 700 :
 710 PROCbold("Bold text")
 720 PRINT '"Not bold"
 730 :
 740 PRINT '"Press ";
 750 PROCbold("SPACE BAR")
 760 PRINT " to reverse display order"
 770 PRINT "     (Note change in diagonal lines)"
 780 PRINT "Press any other key when done"
 790 :
 800 REPEAT : A%=GET : IF A%=32 THEN ?switch=1 : UNTIL A%<>32
 810 :
 820 REM Disable events and return to shadow memory write and
     display
 830 *FX13,4
 840 MODE 128
 850 END
 860 :
 870 DEF PROCdraw(X1,Y1,X2,Y2)
 880 REM A routine that does an approximate job of drawing a
 890 REM line from (X1,Y1) to (X2,Y2) on both screens
 900 REM It is by no means perfect, but will do for a
 910 REM demonstration
 920  LOCAL X1%,X2%,Y1%,Y2%
 930 IF ABS(Y2-Y1)<=2 THEN PROChline : ENDPROC
 940 REM Main memory write
 950 *FX 112,1
 960 Y1%=(Y1+2)/4
 970 Y1%=Y1%*4 : X1%=X1+(X2-X1)*(Y1%-Y1)/(Y2-Y1)
 980 Y2%=(Y2+2)/4
 990 Y2%=Y2%*4 : X2%=X2+(X2-X1)*(Y2%-Y2)/(Y2-Y1)
1000 MOVE X1%,Y1%
1010 DRAW X2%,Y2%
1020 :
1030 REM Shadow memory write
1040 *FX112,2
1050 Y1%=Y1/4
1060 Y1%=Y1%*4+2 : X1%=X1+(X2-X1)*(Y1%-Y1)/(Y2-Y1)
1070 Y2%=Y2/4
1080 Y2%=Y2%*4+2 : X2%=X2+(X2-X1)*(Y2%-Y2)/(Y2-Y1)
1090 MOVE X1%,Y1%
1100 DRAW X2%,Y2%
1110 ENDPROC
1120 : 1130 DEF PROChline
1140 REM For when line is more or less horizontal
1150 LOCAL Y%
1160 Y%=0.5*(Y1+Y2)+0.5
1170 REM Choose which screen it should be in
1180 IF (Y% MOD 4) <2 THEN OSCLI"FX112,1" ELSE OSCLI"FX112,2"
1190 MOVE X1,Y1
1200 DRAW X2,Y2
1210 ENDPROC
1220 :
```

```
1230 DEF PROCbold(A$)
1240 LOCAL X%,Y%
1250 REM print string in both screens, at current position, without
1260 REM a line feed at the end
1270 :
1280 X%=POS : Y%=VPOS
1290 REM write to main memory
1300 *FX112,1
1310 PRINT A$;
1320 REM write to shadow memory
1330 *FX112,2
1340 PRINT TAB(X%,Y%);A$;
1350 REM reset to main memory write
1360 *FX112,1
1370 ENDPROC
```

****

AUCKLAND BULLETIN BOARD

The Auckland User group has advised the following information regarding the Auckland Bulletin Board:

Telephone Number:	(09) 390-836.
Operational details are:	8 Bit data, no parity, 1 stop bit.
Availablity:	24 hours / day.

****

MEMBERSHIP

Membership of the User Group is on payment of an annual subscription from April to March. For the 1986/7 year it is: $30.00, plus $5.00 for Wellington members as a local branch levy, which entitles attendance at meetings as detailed above. Joining or renewing members get all the back issues of the current year - those joining from 1 October may have half the annual sub waived and only receive newsletters from 1 October.

****

USER GROUP MEETINGS

AUCKLAND: -2nd and 3rd Wednesday of each month at Social Club, Auckland Hospital, Grafton Road at 7.30pm. Contact John Buchanan (Secy), 13 Ohinau St, Remuera; David Wilkin (Pres), 29 Rose Camp Rd, Birkdale

North Shore: -last Wednesday of the month, 7.30pm, Salvation Army Hall, Glenfield Road, Glenfield ph Chris 419-0543 (wk)

CHRISTCHURCH: -fortnightly workshops, Monday 6.30pm at Hagley High School. ph. Michael 582-267.

DUNEDIN: -first Thursday of the month. Phone Martin McDowall 42-031

HAWKES BAY: -Alternative months Hastings and Napier. Contact Kendall: Napier 435-624, Major: Napier 447-729, Mitch: Hastings 85-395.

INVERCARGILL: -Phone Evan 330-444 Invercargill.

NELSON: -second Monday of the month, 7.30pm, Nelson Baptist Church, Bridge St. Contact Peter: Richmond 8975, Ken: Nelson 85-150, Richard: Richmond 8489.

PALMERSTON NORTH: -Phone Tom Skinner 83-009(wk)

ROTORUA: -Phone Ian 85-583 or Joan 81-536 (independent group)

TAUPO: -every second Wednesday, 7.00pm. Contact David 84-215 (h), 85-124 (wk).

TAURANGA: -every second Wednesday. ph. Chris & Jane 65-076.

TIMARU: -Phone Lloyd van der Krogt Timaru 61-412.

TOKOROA: -1st Friday of month, 7.30pm, Tokoroa High School. Contacts Don Fraser, 68-813 or Graeme Robertson, 65-346 Tokoroa.

WAIKATO: -first Monday of the month, 7.30pm, Waikato Tech Institute. Ph. Alison or John, Morrinsville 6695.

WELLINGTON: -*usually* second WEDNESDAY of the month, 7.30pm, Royal Society Lecture Theatre, Turnbull St, Thorndon. Other dates advised in Beeblet regional news. (May p.39)

-Workshops: fourth Wednesday of month, other venues. ph. Don 848-235 or Mike 785-437.

OTHER CENTRES: -let us know and we will publish details here.

BEEBLET

CONTRIBUTIONS: -most welcome. Listings should be sent in on tape or disc. (Tape - 2 copies, one at 300 baud please). Please include written explanation of listing; text and graphics only material welcome too.

DEADLINES: - For a particular issue, the last day for material is the last day of the month prior to publication.

ADVERTISING: - Rates are $20 per half page (camera ready), deadline as above.

Copyright (c) BBC/ACORN COMPUTER USER GROUP OF NZ INC.

PUBLISHED: monthly except January and mailed to financial members.
Registered Publication with NZPO.

MEMBERSHIP

For membership details, see page 39.

THE BBC/ACORN COMPUTER USER GROUP OF NZ INC, PO Box 9592, WELLINGTON

Reproduced by SANDIFORD PRINTERS